Assignment 04
(due on 11/19 19:00)

PS4_1.R

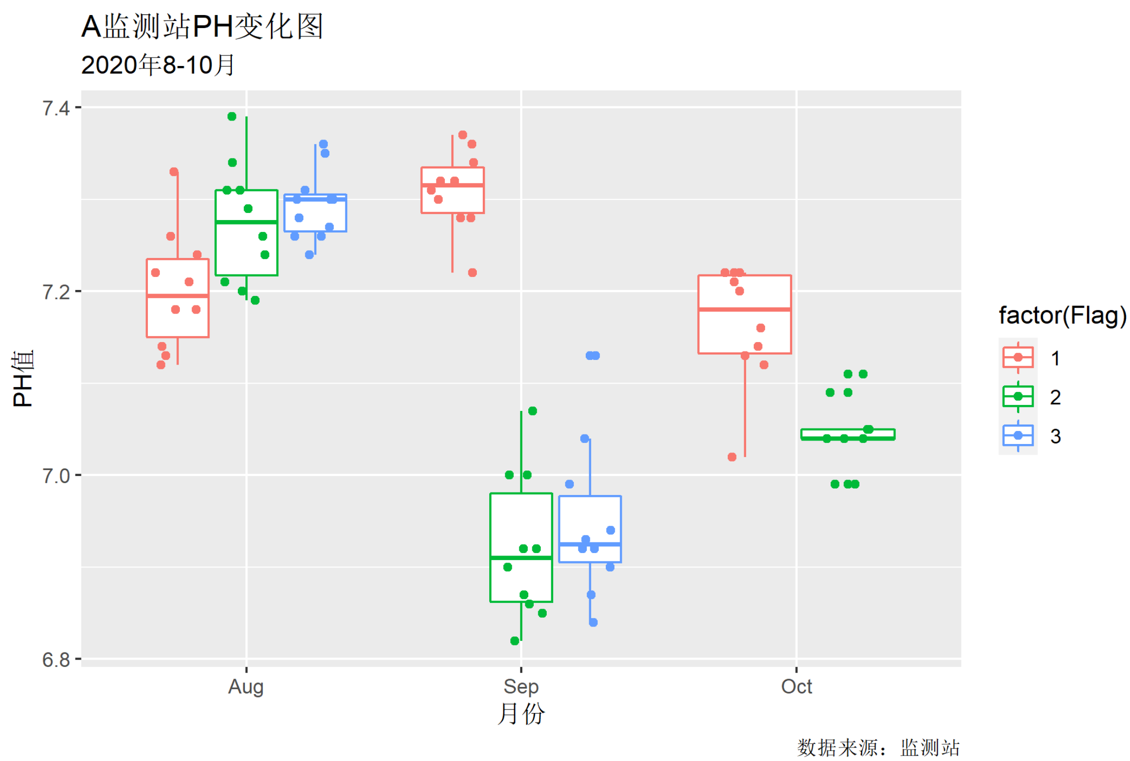
1. Plotting with ggplot2

[25 points] Using research data from your group, make 5 types of plots with the ggplot2 package:

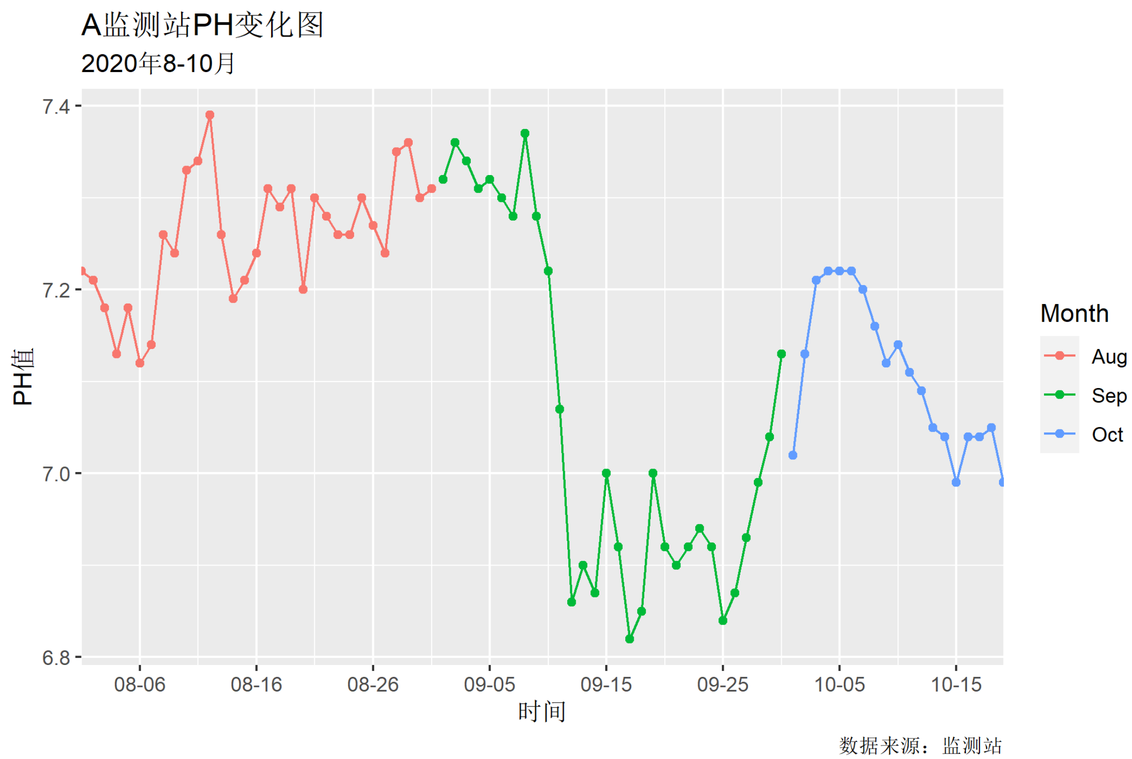
- Boxplot
- Time series
- Histogram
- Scatter plot
- Image plot (you can use data set of interest for this one)

For each one, your plot will be graded from 0 to 5 points based on the number of elements (e.g., aesthetics, legend, panel, axis, title, theme, style, text, annotation, map, …) included and the level of sophistication.

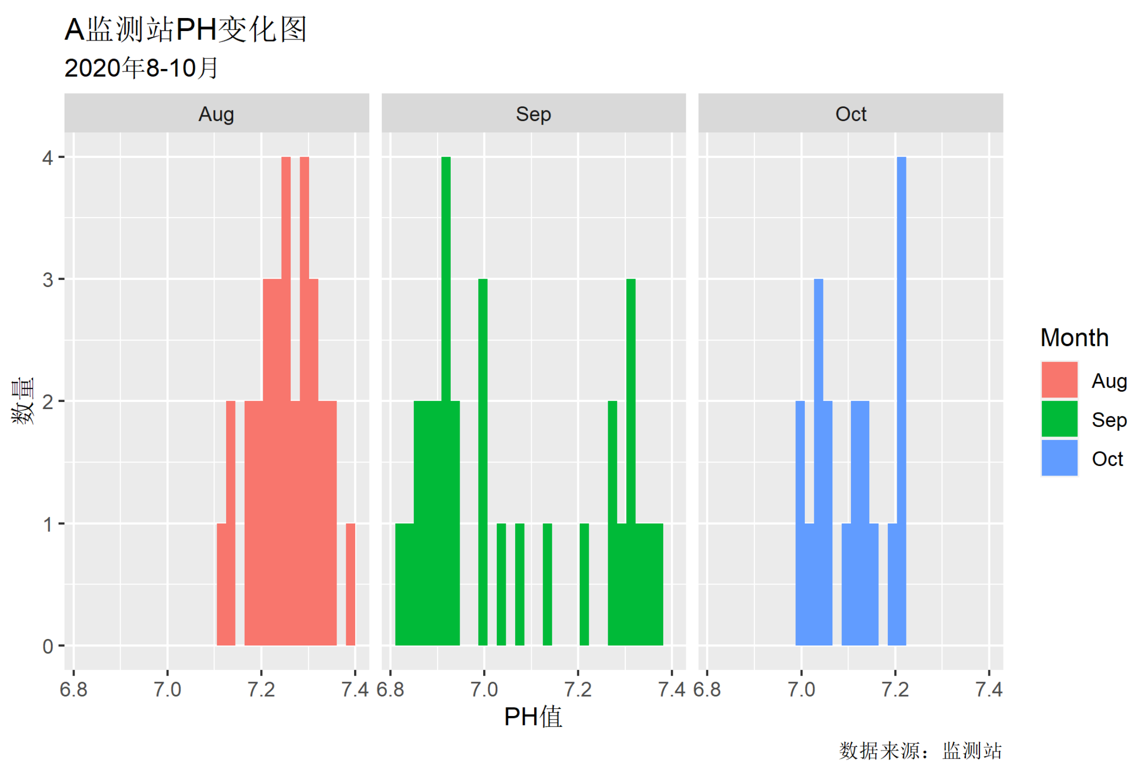
Answer:



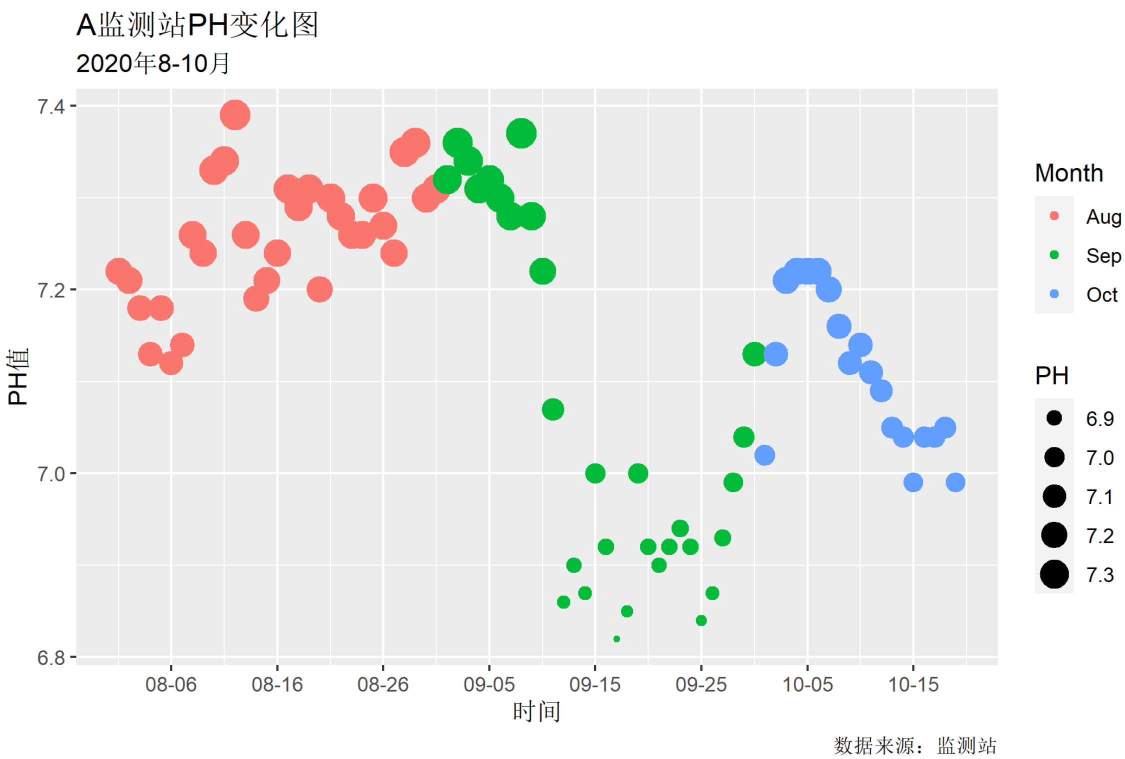
4.1Boxplot



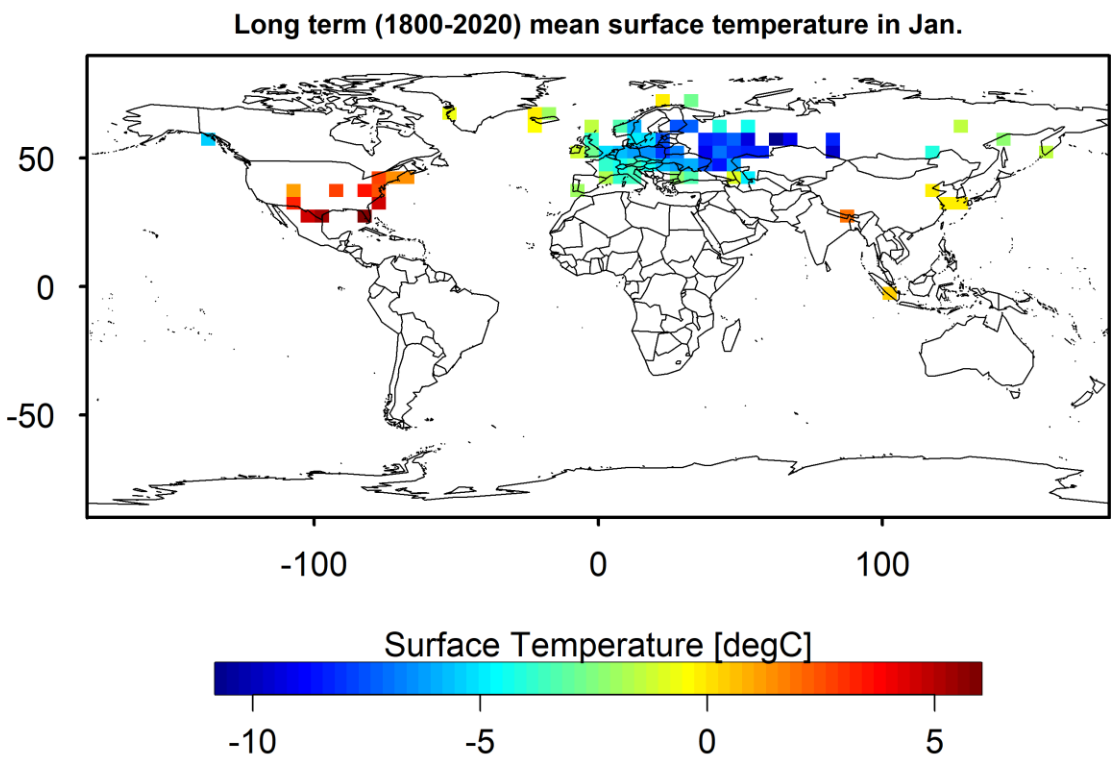
4.2Time series



4.3Histogram



4.4Scatter plot



4.5Image plot

PS4_2.R

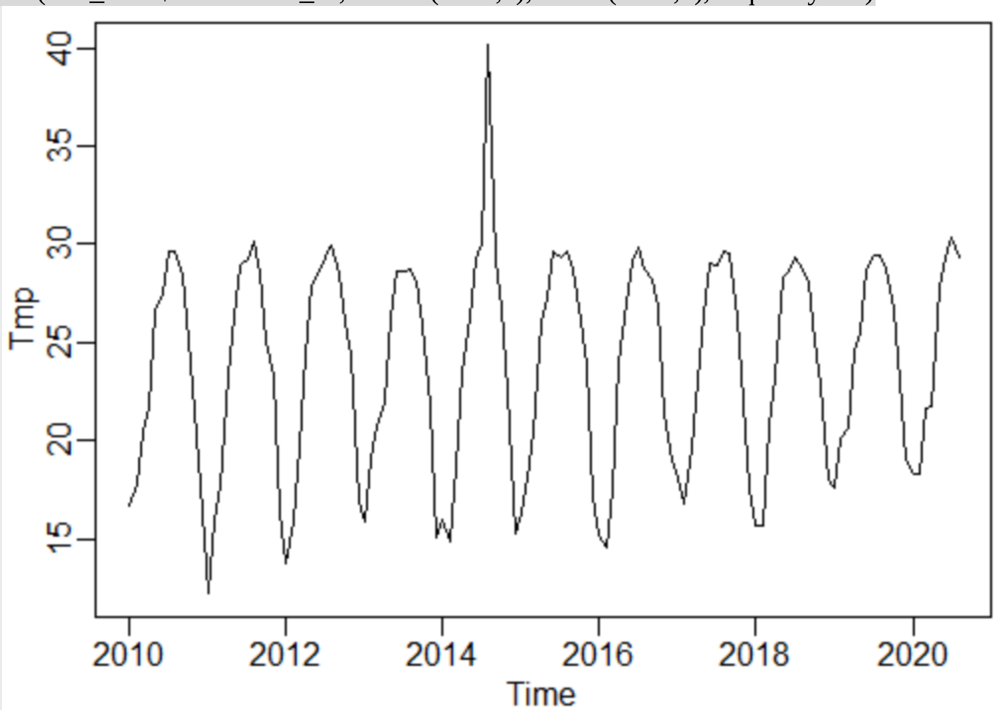
2. Analysis of the time series of monthly temperature

In this exercise, we will take another look at the hourly weather data measured at the BaoAn International Airport during the past 10 years.

2.1 [5 points] Construct a time series of monthly-averaged temperature from 2010 Jan. to 2020 Aug.

Answer:

```
# Get the plot data
Baoan_data <- BaoAn_data_tbl %>%
  select(DATE,TMP) %>%
  mutate(
    Tvalue = as.numeric(substr(TMP,2,5)),
    Tflag = as.logical(as.numeric(substr(TMP,7,7))),
    TMPBaoan = Tvalue * 0.1,
    DATEBaoan = substr(DATE,1,7)) %>%
  filter(TMPBaoan!= 999.9 | Tvalue==TRUE)
#Time = as.Date(DATEBaoan,"%Y-%m")

Plot_Data <- Baoan_data %>%
  select(DATEBaoan,TMPBaoan) %>%
  group_by(DATEBaoan) %>%
  summarise(TMPBaoan_M = mean(TMPBaoan))
head(Plot_Data)

# Apply the ts() function
Tmp <- ts(Plot_Data$TMPBaoan_M, start=c(2010,1), end=c(2020,8),frequency=12)
```

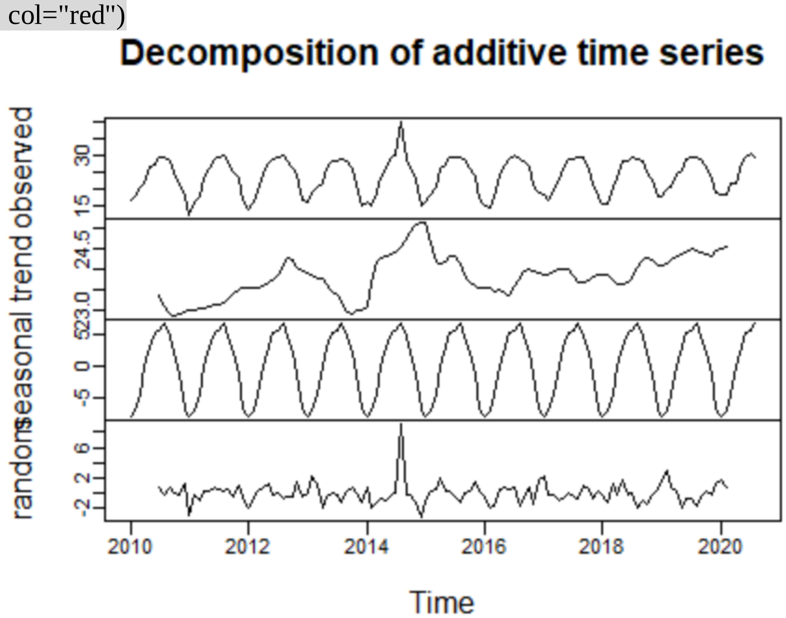


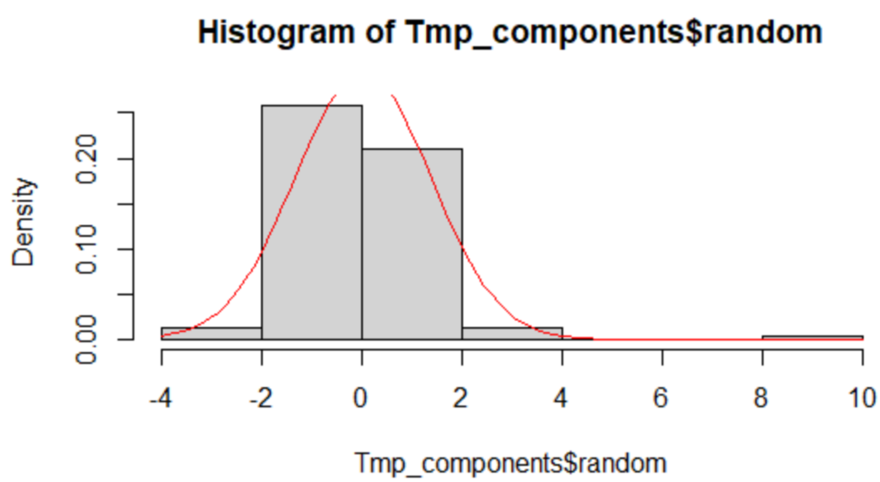
2.2 [5 points] Decompose the time series into trend, seasonality, and error parts. Check whether the error part follows a white noise distribution.

Answer:

As we can see the distribution is a Gaussian white noise, which is a particularly useful white noise series.

```
Tmp_components <- decompose(Tmp)
plot(Tmp_components)

# Plot hist
hist(Tmp_components$random, prob=TRUE)
# Add pdf
curve(dnorm(x, mean=mean(Tmp_components$random,na.rm=T),
        sd=sd(Tmp_components$random,na.rm=T)),
    add=TRUE, col="red")
```



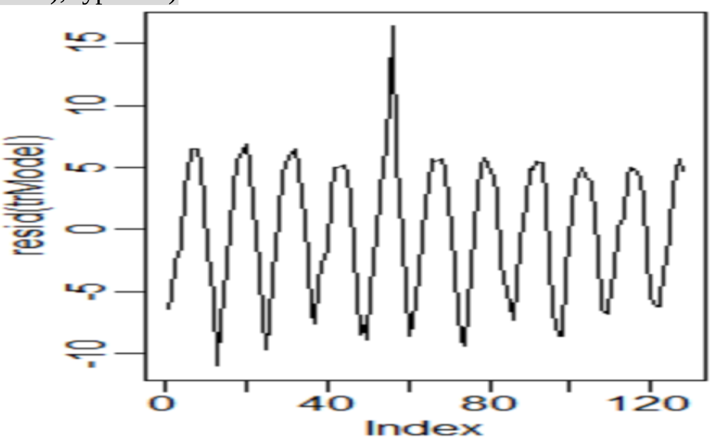


2.3 [10 points] Fit an ARIMA(p,d,q) model to the time series. Describe the fitting process in details in your report.

Answer:

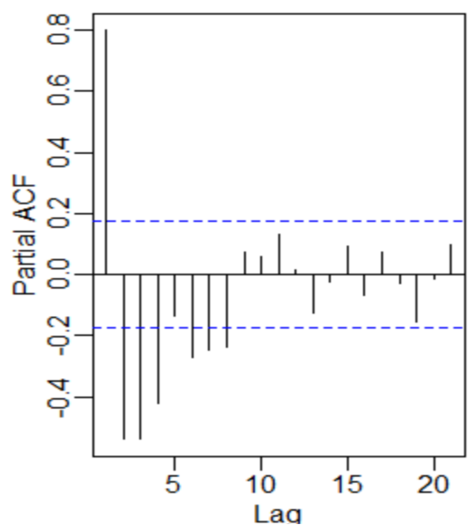
Step1: Through the plot of 2.1 we can find that the data has a small trend, so the first step is to de-trend the data.

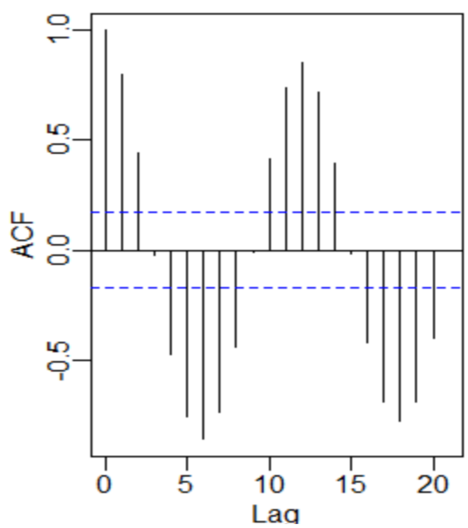
```
trModel <- lm(Tmp ~ c(1:length(Tmp)))
plot(resid(trModel), type="l")
```



Step2: To examine which p and q values will be appropriate we can run acf() and pacf() function.

```
acf(resid(trModel))
pacf(resid(trModel))
```





From ACF and PACF results, we can see that PACF presents a more obvious exponential smoothing trend than ACF. Therefore, we first guess ARIMA model as ARIMA (1,0,0) (2,1,0) [12].

Step3: We can use auto.arima() to obtain the model.

auto.arima(Tmp,trace=T)

```
ARIMA(2,0,2)(1,1,1)[12] with drift        : Inf
 ARIMA(0,0,0)(0,1,0)[12] with drift        : 508.7297
 ARIMA(1,0,0)(1,1,0)[12] with drift        : 472.9121
 ARIMA(0,0,1)(0,1,1)[12] with drift        : Inf
 ARIMA(0,0,0)(0,1,0)[12]                   : 506.9211
 ARIMA(1,0,0)(0,1,0)[12] with drift        : 504.5371
 ARIMA(1,0,0)(2,1,0)[12] with drift        : 461.1239
 ARIMA(1,0,0)(2,1,1)[12] with drift        : Inf
 ARIMA(1,0,0)(1,1,1)[12] with drift        : Inf
 ARIMA(0,0,0)(2,1,0)[12] with drift        : 461.866
 ARIMA(2,0,0)(2,1,0)[12] with drift        : 462.4849
 ARIMA(1,0,1)(2,1,0)[12] with drift        : 463.0189
 ARIMA(0,0,1)(2,1,0)[12] with drift        : 461.6444
 ARIMA(2,0,1)(2,1,0)[12] with drift        : 464.5019
 ARIMA(1,0,0)(2,1,0)[12]                   : 460.0255
 ARIMA(1,0,0)(1,1,0)[12]                   : 471.3721
 ARIMA(1,0,0)(2,1,1)[12]                   : Inf
 ARIMA(1,0,0)(1,1,1)[12]                   : Inf
 ARIMA(0,0,0)(2,1,0)[12]                   : 461.3785
 ARIMA(2,0,0)(2,1,0)[12]                   : 461.1236
 ARIMA(1,0,1)(2,1,0)[12]                   : 461.7545
 ARIMA(0,0,1)(2,1,0)[12]                   : 460.6987
 ARIMA(2,0,1)(2,1,0)[12]                   : 463.1182

 Best model: ARIMA(1,0,0)(2,1,0)[12]

Series: Tmp
ARIMA(1,0,0)(2,1,0)[12]

Coefficients:
         ar1     sar1     sar2
      0.1746  -0.6908  -0.3343
s.e.  0.0926   0.0887   0.0859

sigma^2 estimated as 2.787:  log likelihood=-225.83
AIC=459.67   AICc=460.03   BIC=470.68
```

auto.arima provides two best models ARIMA (1,0,0) (2,1,0) [12], we need to test which is more suitable.

Step4: The coefficients of the two models are significant, while the AIC and BIC of ARIMA (1,0,0) (1,0,0) [12] are bigger than that of ARIMA (1,0,0) (2,1,0) [12]. Therefore, ARIMA(1,0,0) (2,1,0) [12] is selected.

```
airarima1
Call:
arima(x = Tmp, order = c(1, 0, 0), seasonal = list(order = c(1, 0, 0), period = 12),
   method = "ML")
```

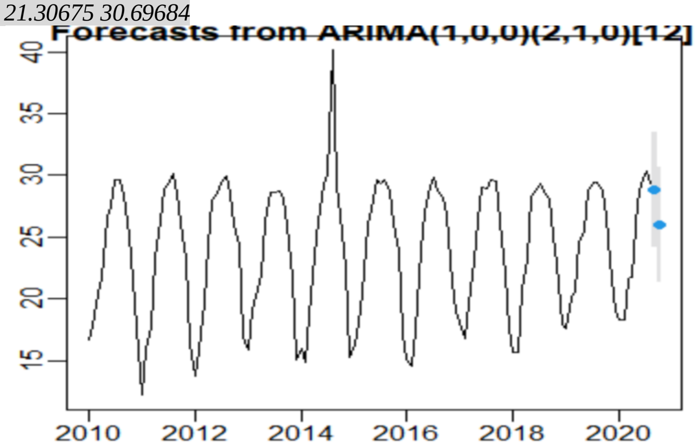
```
Coefficients:
         ar1    sar1   intercept
      0.3981  0.834    23.9936
s.e.  0.0997  0.050     1.2651

sigma^2 estimated as 3.992:  log likelihood = -277.45,  aic = 562.89

> airarima2
Call:
arima(x = Tmp, order = c(1, 0, 0), seasonal = list(order = c(2, 1, 0), period = 12),
    method = "ML")

Coefficients:
         ar1      sar1      sar2
      0.1746  -0.6908  -0.3343
s.e.  0.0926   0.0887   0.0859

sigma^2 estimated as 2.715:  log likelihood = -225.83,  aic = 459.67
```

2.4 [5 points] Predict monthly-averaged temperatures in 2020 Sep. and Oct. with the ARIMA model from 2.3. The predictions will be evaluated against actual observations in those two months.

Answer:

```
Tmpforecast <- forecast(airarima2,h=2,level=c(99.5))
Tmpforecast
forecast:::plot.forecast(Tmpforecast)
```

```
> Tmpforecast
         Point Forecast  Lo 99.5  Hi 99.5
Sep 2020       28.82162 24.19657 33.44667
Oct 2020       26.00180 21.30675 30.69684
```



Time	Tmp_observation	Tmp_forecast	Relative bias
2020/9	29.45206	28.82162	-2.14%
2020/10		26.0018	

The predictions is evaluated against actual observations in September and October, the relative bias is about 2.14%, it's small.